1. Import Libraries

In [2]:

```
import pandas as pd
import matplotlib.pyplot as plt
```

2. Import Data Set

In [4]:

```
data=pd.read_csv('diabetes.csv')
data
```

Out[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunct
0	6	148	72	35	0	33.6	0.
1	1	85	66	29	0	26.6	0.
2	8	183	64	0	0	23.3	0.
3	1	89	66	23	94	28.1	0.
4	0	137	40	35	168	43.1	2.
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	0.
764	2	122	70	27	0	36.8	0.
765	5	121	72	23	112	26.2	0.
766	1	126	60	0	0	30.1	0.
767	1	93	70	31	0	30.4	0.

768 rows × 9 columns

3. Data Undestanding

In [5]:

```
data.isnull().sum()
```

Out[5]:

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [6]:

```
data.dtypes
```

Out[6]:

```
Pregnancies                   int64
Glucose                       int64
BloodPressure                 int64
SkinThickness                 int64
Insulin                       int64
BMI                         float64
DiabetesPedigreeFunction    float64
Age                           int64
Outcome                       int64
dtype: object
```

3. Data Pre Processing

In [17]:

```
from sklearn.preprocessing import StandardScaler
```

In [24]:

```
scaled=StandardScaler()
scaled_x=scaled.fit_transform(x)
scaled_x=pd.DataFrame(scaled_x,columns=x.columns)
scaled_x
```

Out[24]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedig
0	0.639947	0.848324	0.149641	0.907270	-0.692891	0.204013	
1	-0.844885	-1.123396	-0.160546	0.530902	-0.692891	-0.684422	
2	1.233880	1.943724	-0.263941	-1.288212	-0.692891	-1.103255	
3	-0.844885	-0.998208	-0.160546	0.154533	0.123302	-0.494043	
4	-1.141852	0.504055	-1.504687	0.907270	0.765836	1.409746	
...	...	...	...	...	...	...	
763	1.827813	-0.622642	0.356432	1.722735	0.870031	0.115169	
764	-0.547919	0.034598	0.046245	0.405445	-0.692891	0.610154	
765	0.342981	0.003301	0.149641	0.154533	0.279594	-0.735190	
766	-0.844885	0.159787	-0.470732	-1.288212	-0.692891	-0.240205	
767	-0.844885	-0.873019	0.046245	0.656358	-0.692891	-0.202129	

768 rows × 8 columns

In [25]:

```
x=data.drop('Outcome',axis=1)
y=data[['Outcome']]
```

In [30]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(scaled_x,y,test_size=0.20,shuffle=True,rando
```

Hyper Parameter

5.Model Building

In [78]:

```
from keras.layers import Dense,Flatten
from keras.models import Sequential
from sklearn.model_selection import GridSearchCV
from keras.wrappers.scikit_learn import KerasClassifier
import tensorflow as tf
import warnings
warnings.filterwarnings('ignore')
from sklearn.model_selection import KFold
```

In [90]:

```
    model=Sequential()
    model.add(Dense(units=10,input_dim=8,kernel_initializer='glorot_normal',activation="rel
    model.add(Dense(units=8,input_dim=8,kernel_initializer='glorot_normal',activation="relu
    model.add(Dense(units=1,activation='sigmoid'))
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),loss='binary_cros
    model.fit()
```

In [69]:

```
def model_fun():
    model=Sequential()
    model.add(Dense(units=10,input_dim=8,kernel_initializer='glorot_normal',activation="rel
    model.add(Dense(units=8,input_dim=8,kernel_initializer='glorot_normal',activation="relu
    model.add(Dense(units=1,activation='sigmoid'))
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),loss='binary_cros
    return model
```

In [84]:

```
keras_model=KerasClassifier(build_fn=model_fun)
```

In [86]:

```
kfold=KFold(n_splits=5,shuffle=True,random_state=1)
```

In [91]:

```
id={'batch_size':[10,20,30],'epochs':[10,20,30],'steps_per_epoch':[10,20,30],'validation_ba
```

```
10/10 [==============================] - 0s 2ms/step - loss: 0.7113 - bina
ry_accuracy: 0.5900
Epoch 4/10
10/10 [==============================] - 0s 2ms/step - loss: 0.7117 - bina
ry_accuracy: 0.6600
Epoch 5/10
10/10 [==============================] - 0s 2ms/step - loss: 0.6975 - bina
ry_accuracy: 0.6413
Epoch 6/10
10/10 [==============================] - 0s 2ms/step - loss: 0.7521 - bina
ry_accuracy: 0.5700
Epoch 7/10
10/10 [==============================] - 0s 2ms/step - loss: 0.6127 - bina
ry_accuracy: 0.6800
Epoch 8/10
10/10 [==============================] - 0s 2ms/step - loss: 0.6241 - bina
ry_accuracy: 0.6400
Epoch 9/10
10/10 [==============================] - 0s 2ms/step - loss: 0.6882 - bina
ry_accuracy: 0.6000
```

In [92]:

```
grid.best_params_
```

Out[92]:

```
{'batch_size': 10,
 'epochs': 10,
 'steps_per_epoch': 10,
 'validation_batch_size': 10}
```